IN THE CLAIMS:

Please cancel claims 39-47 without prejudice.

Please amend claims 1, 20, 22-23, 26-32 and 37 to better claim the invention.

1   1. (Currently Amended) A method for converting a file access data structure from a first

2   endianness to a second endianness by a processor, the method comprising the steps of:

3   determining if the file access data structure is a critical path data structure, where

4   the critical path data structure includes commonly utilized data structures;

5   in response to determining that the file access data structure is the critical path

6   data structure, performing byte swap operations using specific code functions and placing

7   a converted data structure in an output buffer to make it available for further processing;

8   in response to determining that the file access data structure is not the critical path

9   data structure, calling a byte-swapping engine to perform the following:

10   a.) identifying, from a descriptor look up table, a series of actions to per-

11   form on elements of the file access data structure , where the series of actions in-

12   clude at least one of converting, copying, or linking ; and

13   b.) performing the identified series of actions on the elements of the file

14   access data structure to convert the file data structure from the first endianness to

15   the second endianness; and

16   c.) placing a converted data structure in an output buffer to make it avail-

17   able for further processing.

2.-19.   (Cancelled)

1   20.   (Currently Amended) A method for converting a data structure by a processor, com-

2   prising:

3   determining if the data structure is a critical path data structure, where the critical

4   path data structure includes commonly utilized data structures;

5   in response to determining that the data structure is the critical path data structure,

6   performing byte swap operations using specific code functions and placing a converted

7   data structure in an output buffer to make it available for further processing and placing a

8   converted data structure in an output buffer to make it available for further processing;

9   in response to determining that the data structure is not the critical path data struc-

10   ture, calling a byte-swapping engine and performing the following: ;

11   a.) providing a file access data structure as input to the byte-swapping en-

12   gine;

13   b.) providing a descriptor look up table to the byte-swapping engine;

14   c.) identifying, from the descriptor look up table, a series of actions to per-

15   form on elements of the file access data structure in order to swap bytes of the file

16   access data structure from a first endianness to a second endianness ~~, where the~~

17   ~~series of actions include at least one of converting, copying, or linking~~ ; and

3

18   d.) performing the identified series of actions on the elements of the file

19   access data structure to convert the file access data structure; and

20   e.) placing a converted data structure in an output buffer to make it avail-

21   able for further processing .

1   21. (Previously Presented) The method as in claim 20, further comprising:

2   using as the file access data structure a file having Direct Access File System

3   (DAFS) protocol.

1   22. (Currently Amended) ~~The method as in claim 20, further comprising:~~

2   A method for converting a data structure by a processor, comprising:

3   determining if the ~~file access~~ data structure is a critical path data structure, where

4   the critical path data structure includes commonly utilized data structures ; ~~, and~~

5   in response to determining that ~~if~~ the ~~file access~~ data structure is ~~a~~ the critical path

6   data structure, performing byte swap operations using specific code functions; and

7   placing a converted data structure in an output buffer to make it available for fur-

8   ther processing .

1   23. (Currently Amended) ~~The method as in claim 20, further comprising:~~

2   A method for converting a data structure by a processor, comprising:

3   determining if the ~~file access~~ data structure is a critical path data structure, where

4   the critical path data structure includes commonly utilized data structures ; ~~, and~~

5   in response to determining that the data structure is a critical path data structure,

6   performing byte swap operations using specific code functions and placing a converted

7   data structure in an output buffer to make it available for further processing;

8   in response to determining that the ~~if the file access~~ data structure is not ~~a~~ the

9   critical path data structure, ~~perform byte swap operations on a data structure header~~ call-

10   ing a byte-swapping engine and performing the following:

11   a.) providing a file access data structure as input to the byte-swapping en-

12   gine;

13   b.) providing a descriptor look up table to the byte-swapping engine;

14   c.) identifying, from the descriptor look up table, a series of actions to per-

15   form on elements of the file access data structure in order to swap bytes of the file

16   access; and

17   d.) performing the identified series of actions on the elements of the data

18   structure header to convert the file access data structure; and

19   e.) placing a converted data structure in an output buffer to make it avail-

20   able for further processing .

1   24. (Previously Presented) The method as in claim 20, further comprising:

2   swapping bytes of the data structure as needed, in response to swapping bytes of

3   the file access data structure.

1   25. (Previously Presented) The method as in claim 20, further comprising:

2   determining if an element entry of the descriptor look up table is nested;

3   branching to the nested entry;

4   identifying, from the descriptor look up table, a nested series of actions to perform

5   on elements of the nested entry in order to swap bytes of the entry from a first endianness

6   to a second endianness, where the nested series of actions includes linking and convert-

7   ing.

1   26. (Currently Amended) A computer to convert a data structure by a processor, com-

2   prising:

3   means for determining if the data structure is a critical path data structure, where

4   the critical path data structure includes commonly utilized data structures;

5   means for ~~calling a byte-swapping engine~~ performing byte swapping operations

6   using specific code functions in response to determining that the data structure is the

7   critical path data structure; and

8   ~~means for providing a file access data structure as input to the byte-swapping en-~~

9   ~~gine;~~

10   ~~means for providing a descriptor look-up table to the byte-swapping engine;~~

11   ~~means for identifying, from the descriptor look-up table, a series of actions to per-~~

12   ~~form on elements of the file access data structure in order to swap bytes of the file access~~

13   ~~data structure from a first endianness to a second endianness~~ , where the series of actions

14   ~~include at least one of converting, copying, or linking~~ ; and

15   ~~means for performing the identified series of actions on the elements of the file~~

16   ~~access data structure to convert the file access data structure~~

17   means for placing a converted data structure in an output buffer to make it avail-

18   able for further processing .

1   27. (Currently Amended) The computer as in claim 26, further comprising:

2   means for using as the ~~file access~~ data structure a file having Direct Access File

3   System (DAFS) protocol.

6

1 28. (Currently Amended) ~~The computer as in claim 26, further comprising:~~

2 A computer to convert a data structure by a processor, comprising:

3 means for determining if the ~~file access~~ data structure is a critical path data struc-

4 ture, where the critical path data structure includes commonly utilized data structures; ~~,~~

5 means for ~~and if the file access data structure is a critical path data structure,~~ per-

6 form~~ing~~ byte swap operations using specific code functions in response to determining

7 that the data structure is the critical path data structure, and placing a converted data

8 structure in an output buffer to make it available for further processing; ~~means for per-~~

9 ~~forming the identified series of actions on the elements of the file access data structure to~~

10 ~~convert the file access data structure~~

11 means for calling a byte-swapping engine performing the following in response to

12 determining that the data structure is not the critical path data structure:

13 a.) means for providing a file access data structure as input to the byte-

14 swapping engine;

15 b.) means for providing a descriptor look up table to the byte-swapping

16 engine; and

17 c.) means for identifying, from the descriptor look up table, a series of ac-

18 tions to perform on elements of the file access data structure in order to swap

19 bytes of the file access data structure from a first endianness to a second endian-

20 ness; and

21 d.) means for placing a converted data structure in an output buffer to

22 make it available for further processing .

7

1   29. (Currently Amended) ~~The computer as in claim 26, further comprising:~~

2   <u>A computer to convert a data structure by a processor, comprising:</u>

3   means for determining if the ~~file access~~ data structure is a critical path data struc-

4   ture, where the critical path data structure includes commonly utilized data structures <u>:</u> ~~;~~

5   <u>means for calling a byte swapping engine and performing the following in re-</u>

6   <u>sponse to determining that</u> the ~~and if the file access~~ data structure is not ~~a~~ <u>the</u> critical path

7   data structure<u>:</u> ~~, perform byte swap operations on a data structure header~~

8   <u>means for providing a file access data structure as input to the byte-swapping en-</u>

9   <u>gine;</u>

10   <u>means for providing a descriptor look up table to the byte-swapping engine;</u>

11   <u>means for identifying, from the descriptor look up table, a series of actions to per-</u>

12   <u>form on elements of the data structure header in order to swap bytes of the file access</u>

13   <u>data structure from a first endianness to a second endianness; and</u>

14   <u>means for placing a converted data structure in an output buffer to make it avail-</u>

15   <u>able for further processing</u> .

1   30. (Currently Amended) The computer as in claim ~~26~~ <u>29</u>, further comprising:

2   means for swapping bytes of the data structure as needed, in response to swapping

3   bytes of the file access data structure.

1   31. (Currently Amended) The computer as in claim ~~26~~ <u>29</u>, further comprising:

2   means for determining if an element entry of the descriptor look up table is

3   nested;

4   means for branching to the nested entry; and

5   means for identifying, from the descriptor look up table, a nested series of actions

6   to perform on elements of the nested entry in order to swap bytes of the entry from a first

7   endianness to a second endianness, where the nested series of actions includes converting

8   and linking.

1   32. (Currently Amended) A computer readable media, comprising:

2   said computer readable media containing instructions for execution on a processor

3   for the practice of a method for converting a data structure by a processor, the method

4   having the steps of,

5   determining if the data structure is a critical path data structure, where the critical

6   path data structure includes commonly utilized data structures;

7   in response to determining that the data structure is the critical path data structure,

8   performing byte swap operations using specific code functions and placing a converted

9   data structure in an output buffer to make it available for further processing;

10   in response to determining that the data structure is not the critical path data struc-

11   ture, calling a byte-swapping engine and performing the following: ;

12   a.) providing a file access data structure as input to the byte-swapping en-

13   gine;

14   b.) providing a descriptor look up table to the byte-swapping engine;

15   c.) identifying, from the descriptor look up table, a series of actions to per-

16   form on elements of the file access data structure in order to swap bytes of the file

17   access data structure from a first endianness to a second endianness ~~where the se-~~

18   ~~ries of actions include at least one of converting, copying, or linking~~; and

19   d.) performing the identified series of actions on the elements of the file

20   access data structure to convert the file access data structure; and

21   e.) placing a converted data structure in an output buffer to make it avail-

22   able for further processing .

1   33.-36. (Cancelled)

1   37. (Currently Amended) A method for converting a first data structure ~~from a~~ to a sec-

2   ond data structure by a processor, the method comprising the steps of:

3   determining if the first data structure is a critical path data structure, where the

4   critical path data structure includes commonly utilized data structures;

5   in response to determining that the first data structure is not the critical path data

6   structure, calling a byte-swapping engine;

7   using a descriptor lookup table, by the byte-swapping engine, to provide actions

8   to be performed on each element of the first data structure; ~~and~~

9   stepping through the descriptor table, by the byte-swapping engine, and process-

10   ing each element of the first data structure according to the element's size and action to

11 convert the first data structure <u>having an element with a first endianness</u> into the second

12 data structure <u>having an element with a second endianness; and</u>

13 <u>placing the second data structure in an output buffer to make it available for fur-</u>

14 <u>ther processing</u> .

1 38. (Previously Presented) The method of claim 37, further comprising:

2 using a byte as the data structure.

39 - 47 (Cancelled)

Please add new claims 48, *et seq.*

1 48. (NEW) The method of claim 37, further comprising:

2 determining if the first data structure is a critical path data structure, and if it is a

3 critical path data structure then converting an element of the first data structure from a

4 first endianness to a second endianness using a set of specific code functions.

1 49. (NEW) The method of claim 48, further comprising:

2 determining that the critical data path structure is a direct access file system

3 (DAFS) header data structure.

1 50. (NEW) The method of claim 48, further comprising:

2 designing the specific code functions to rapidly convert any elements of the data

3 structure to the second endianness without using a byte swapping engine

1 51. (NEW) A computer to convert a file access data structure from a first endianness to

2 a second endianness, comprising:

3 an operating system executing on the computer to determine if the file access data

4 structure is a critical path data structure, where the critical path data structure includes

5 commonly utilized data structures;

6 the computer performing byte swap operations using specific code functions in

7 response to determining that the file access data structure is the critical path data struc-

8 ture, and holding a converted data structure in an output buffer to make it available for

9 further processing;

10 the computer calling a byte-swapping engine to perform the following in response

11 to determining that the file access data structure is not the critical path data structure:

12 a.) the byte-swapping engine executing a process to identify, from a de-

13 scriptor look up table, a series of actions to perform on elements of the file access

14 data structure; and

15 b.) the byte-swapping engine to perform the identified series of actions on

16 the elements of the file access data structure to convert the file data structure from

17 the first endianness to the second endianness; and

18   c.) a memory having an output buffer to hold a converted data structure in

19   the output buffer to make it available for further processing.

1   52. (NEW) The computer as in claim 51, further comprising:

2   an operating system executing on the computer to use as the file access data struc-

3   ture a file having Direct Access File System (DAFS) protocol.

1   53. (NEW) The computer as in claim 51, further comprising:

2   the critical path data structures are small data structures.

1   54. (NEW) The computer of claim 51, further comprising:

2   the operating system to determine that the critical data path structure is a direct

3   access file system (DAFS) header data structure.

1   55. (NEW) The computer of claim 51, further comprising:

2   the operating system to determine to execute the specific code functions to rapidly

3   convert any elements of the data structure to the second endianness without using a byte

4   swapping engine.

1   56. (NEW) The computer as in claim 51, further comprising:

2   the operating system executing on the computer to swap bytes of a file as needed,

3   in response to swapping bytes of the file access data structure.

1   57. (NEW) The computer as in claim 51, further comprising:

2   the operating system executing on the computer to determine if an element entry

3   of the descriptor look up table is nested;

4   the operating system to branch to the nested entry;

5   the operating system to identify, from the descriptor look up table, a nested series

6   of actions to perform on elements of the nested entry in order to swap bytes of the entry

7   from a first endianness to a second endianness, where the nested series of actions in-

8   cludes linking and converting.

1   58. (NEW) the method of claim 1, further comprising:

2   the critical path data structures are small data structures.